Towards Asynchronous Group Recommendations

Ondřej Kříž[1], Patrik Dokoupil[1] and Ladislav Peška[1,*]

[1]Faculty of Mathematics and Physics, Charles University, Prague, Czechia

Abstract

Group recommender systems are often studied under the assumption that all group members interact with the system synchronously through a shared interface. That is, all group members observe the same recommendations at the same time, e.g., on a shared display. However, many real-world decision scenarios involve users who are physically distributed and interact with other group members through their personal mobile devices. This opens an avenue towards asynchronous group recommendations, where each user may receive a different set of recommendations at the given time, while maintaining the overarching goal of making a well-informed group decision. While quite natural in its application, this research direction remains largely unaddressed in related work. In this foundational contribution, we define the (semi-)asynchronous group recommendation task, propose a recommendation framework instantiating it, and compare it against synchronous baselines in an offline simulation evaluation focused on time to reach consensus. Code & raw results are available at https://github.com/Cross-bit/group-consensus-eval.

Keywords

Group recommender systems, Group decision making, Asynchronous recommendation to groups

1. Introduction

Recommender systems (RS) are essential for navigating modern online platforms and nowadays represent an irreplaceable component on many large-scale services such as Netflix, LinkedIn, or Spotify. Nevertheless, while most such services target individual users and their needs in their recommendations, many daily decisions are made collectively in a group of users. Some common examples are selecting a restaurant for a team lunch, a movie to watch with friends, or a holiday to pick with a family.

Classical group recommender systems (GRS, [1]) attempt to facilitate this group decision task by generating recommendations that consider all members' preferences simultaneously. These recommendations are being shared with all group members to ease the decision-making process. Classical GRS implicitly assumes a synchronous setting, where all group members see the same recommendations at the same time (i.e., a "single-point-of-entry" setting, such as a shared display [2]). However, this assumption rarely holds in practice. Users nowadays often reach decisions asynchronously through informal, distributed channels like messaging apps, where users exchange ideas until they converge on an agreement. While some approaches [3, 4] have been proposed to support decision-making in similar scenarios, these are mostly focused on group sentiment analysis and reaching a final group choice, rather than on the earlier stages of identifying viable candidate items.

To address this challenge, we need to first observe that in these distributed settings (i.e., "multiple-points-of-entry"), it is not strictly necessary that all users receive the same recommendations at a given time. Instead, each user may receive a different collection of suggested items at different time points and provide their feedback asynchronously. This requires a central entity, i.e., an asynchronous group recommender system, that orchestrates the suggestions provided to each user based on her preferences and on feedback from other group members. Classical (synchronous) GRS may be used in this role as well. Nevertheless, the compromises made in the process of delivering a unified list of recommendations may render it uninspirational and less desirable for some group members.

Joint Proceedings of the ACM UMAP Workshops 2026, UMAP 2026, June 8–11, 2026, Gothenburg, Sweden

*Corresponding author.

ondrej.kriz875@student.cuni.cz (O. Kříž); patrik.dokoupil@matfyz.cuni.cz (P. Dokoupil); ladislav.peska@matfyz.cuni.cz (L. Peška)

0009-0000-5100-7948 (O. Kříž); 0000-0002-1423-628X (P. Dokoupil); 0000-0001-8082-4509 (L. Peška)

Instead, in this work, we focus on recommending solutions that can leverage the asynchronous multi-step nature of the group decision process. In particular, our solution combines recommendations best suited to each individual user with a redistribution unit that proposes candidates already deemed suitable by a portion of the group to the remaining members. As such, we simultaneously focus on both group-wide exploration of potential candidates and exploitation, i.e., expediting the processing of the most promising candidates throughout the group.

In the rest of this work, we first define the (fully) asynchronous and semi-asynchronous group recommendation tasks that both refer to slightly different group decision processes. Then we propose a generic recommendation framework suitable for semi-asynchronous tasks, and instantiate it with five particular variants. Next, we propose a simulation framework for comparing recommendation algorithms under semi-asynchronous conditions and, finally, compare the proposed solutions against synchronous GRS baselines.

2. Synchronicity and Group Recommendation Tasks

Let us first briefly formalize how the recommendation tasks may differ depending on how the group is expected to handle the decision process.

Fully synchronous task. In the classical GRS setup, all group members $\{u_1, ..., u_k\} \in G$ are assumed to be present at the spot and are expected to use a single device to display the recommendations. Upon receiving the recommendations, users engage in an in-person decision-making process (outside the supporting software), eventually reaching the best possible decision as quickly as possible.[1] This decision may be transparent to the system in some cases and used for future model training.

As such, the recommendation task is, based on the preferences of the whole group G, and the available candidate items $C \subseteq I$, to deliver a single ordered list of recommended items to be displayed to the whole group, i.e. $RS(G, C) \rightarrow L_G$. Requests for additional recommendations may be considered a simple continuation of L_G, as no new feedback is expected during the decision process.

Fully asynchronous task. In contrast to the synchronous setup, the fully asynchronous task assumes that group members interact remotely, using a shared communication platform on which a recommendation service may reside. Individual users join and leave the decision-making process at their discretion, may need to access the recommendations at any given time point, and their intermediate responses are transparent to the system. As the goal of reaching the best possible decision as quickly as possible has not changed, the system should, whenever the request arrives, provide recommendations that maximize the probability of that outcome, given the current stage of the decision-making process.

As such, the recommender system delivers individualized recommendations to each user $u \in G$ based on the current group context (i.e., the group itself and the current decision-making context $X^{[t]}$, where $[t]$ denotes the current timestamp); formally, $RS(u, G, C, X^{[t]}) \rightarrow L_u^{[t]}$. While the decision-making context may be formalized in numerous ways, it is safe to assume that we can construct a projection towards a member-candidate acceptability matrix, based on which the system may identify different convergence criteria, e.g., signaling a "*Match!*" if all group members express a sufficiently positive preference over some candidate item.

Semi-asynchronous task. While the fully asynchronous task is practically relevant, it poses several challenges for both the decision-making process itself and the supporting RS. The varying levels of engagement among group members make it difficult for both the group and the RS to avoid being dominated by the most active or vocal participants, potentially biasing the outcome toward a subset of users. At the same time, the lack of coordination complicates the definition of convergence, as different users may be at different stages of the decision-making process.

[1] Particular definitions and importance of both speed and decision quality criteria may differ across particular situations.

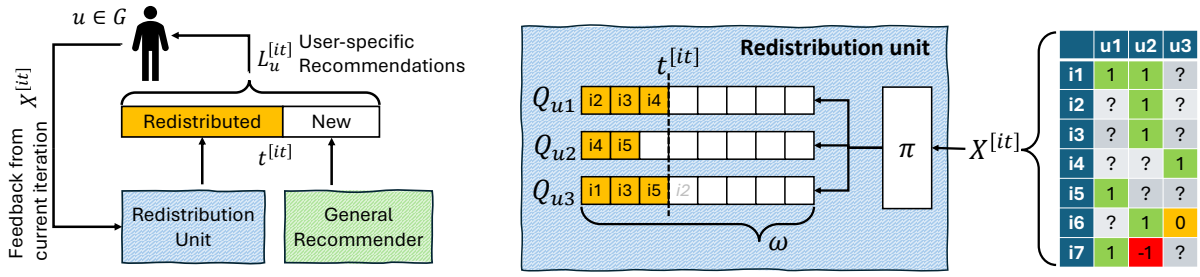

Figure 1: Semi-asynchronous matching framework. Left: high-level architecture, right: detail of the redistribution unit. New candidate items are generated by the general RS, while positively evaluated items are redistributed among other group members through the redistribution unit. Maximal volume of redistributed items is determined by the parameter $t^{[it]}$, while the ordering of redistributed items for individual users is govern by a ranking strategy π favoring items with more positive feedback and higher estimated relevance for the target user. Note that while the item $i2$ can be shown to user $u3$, it was ranked beyond the maximal redistribution threshold $t^{[it]}$ and therefore not shown. In contrast, only two suitable items, $i4$ and $i5$, exist for user $u2$. As such, the redistribution quota has not been filled and additional suggestion from the general recommender is shown instead. Items $i6$ and $i7$ exceeded the maximal acceptable volume of negative or neutral feedback τ_{neg} and therefore not being displayed to any users.

To mitigate these issues, we introduce the *semi-asynchronous task*, which represents a middle ground between the previous settings. In this setup, users engage in remote decision-making via a communication platform with a recommendation overlay, but they are available at approx. the same time and willing to engage in a partially structured decision-making process. The decision process is organized into separate iterations followed by a synchronization check. At each iteration *it*, each group member $u \in G$ receives an individualized list of candidate items $RS(u, G, C, X^{[it-1]}) \rightarrow L_u^{[it]}$. Users then provide feedback for the list, instantiating $X^{[it]}$. When all users complete the evaluation of the current iteration,[2] the system checks whether some convergence criteria have been met, or the next round is administered.

This formulation introduces lightweight coordination into the process, ensuring that all group members progress through the decision-making in a balanced manner, while still preserving personalization within each round. As such, it prevents the decision from being skewed by a subset of highly active users while also enabling the recommender system to periodically reassess group-level agreement and steer the process towards convergence.

3. Semi-asynchronous Group Recommendation Framework

Building on the above-defined semi-asynchronous task, let us now focus on how to instantiate a recommending service for it. To this end, we propose a recommendation framework centered around a single decision-making event *e*. For this event, a fixed group of users $u \in G$ participates in the decision process, which is separated into several iterations *it*. At each iteration, each participant receives a personalized batch of $\omega = |L_u^{[it]}|$ candidate items, and can provide simple feedback over the individual candidates $r(u, i); i \in L_u^{[it]}$.[3] The batch (see Figure 1) consists of two complementary parts: *new* items generated by a *general recommender*, and *redistributed* items drawn from a priority queue Q_u of candidate items already seen by some other group members, but not yet seen by u.

The *redistribution unit* for user u is initialized by items preferred by some $\bar{u} \in G$, but unseen by u. Then, items for which the volume of negative or neutral feedback exceeds the allowable quota $(r_{neg+neutr}(G, i)^{[it]} \geq \tau_{neg})$ are filtered out.[4] Then, the remaining items in Q_u are ranked by priority

[2]Some timeouts are to be expected in production settings.

[3]In practical realization, we opted for a mobile-first environment, where users can provide *positive*, *neutral*, and *negative* feedback via simple swipe gestures. As such, $\forall i \in L_u^{[it]}$, we receive an explicit feedback $r(u, i) \in \{-1, 0, 1\}$.

[4]In case of longer sessions with difficult decisions, this behavior may be further refined and, e.g., let users to re-visit and possibly re-evaluate items initially considered as neutral.

Table 1
Overview of evaluated recommender variants. The *no feedback* variants keep the general recommendation model fixed throughout the session, ignoring any new feedback provided in individual iterations. In contrast, EMA-based variants update the group profile incrementally using feedback obtained in previous iterations. EMA denotes an exponential moving average that gives more weight to recent interactions. *Static* and *dynamic* indicate whether the redistribution threshold $t^{[it]}$ remains fixed or follows the sigmoid schedule.

Short name	Family	Key characteristics
Async-Static-Ind	Semi-Asynchronous	static $t^{[it]}$, individual RS (EASE^R)
Async-Static-Grp	Semi-Asynchronous	static $t^{[it]}$, group RS (AVG)
Async-Dyn-Ind	Semi-Asynchronous	dynamic $t^{[it]}$, individual RS (EASE^R)
Hybrid-Ind	Hybrid	first round synchronous + individual RS + no feedback
Hybrid-Grp-EMA	Hybrid	first round synchronous + group RS + EMA feedback
Sync	Synchronous	no feedback
Sync-EMA	Synchronous	EMA feedback on group profile

function π as follows:

$$\pi(u,i)^{[it]} = r_{pos}(G,i)^{[it-1]} \cdot \hat{r}(u,i), \tag{1}$$

where $r_{pos}(G,i)^{[it-1]}$ denote the number of positive votes item i has received up to iteration $[it-1]$, and $\hat{r}(u,i)$ is the estimated relevance of item i for user u. This ensures that the most promising candidates, i.e., those liked by many group members and seemingly appealing to the current user, are shown first.

The ratio between redistributed and new items is controlled by a parameter $t^{[it]} \in (0,\omega)$, where at most $t^{[it]}$ items are redistributed and $\omega - t^{[it]}$ are newly generated. The redistribution threshold may be kept *static*, but to mimic a classical *first explore then refine* search paradigm [5], the $t^{[it]}$ parameter may also *dynamically* change from more exploration in early iterations towards more redistribution later on. To facilitate this dynamicity, we utilize a modified sigmoid function with a lower bound c_0, initialization shift s, and steepness α as follows:

$$\sigma_{c_0}(k) = \left(c_0 + (1-c_0)\frac{1}{1+e^{-\alpha(it-s)}}\right). \tag{2}$$

Then, the redistribution threshold can be defined as $t^{[it]} = \lceil \omega \cdot \sigma_{c_0}(it) \rceil$.

The *general recommender* component can be instantiated as an arbitrary individual recommender – generating candidates independently for each participant based on their personal profile. Nevertheless, using a group recommender combining the needs of multiple group members is also plausible.[5] In our experiments, we utilized a well-known linear autoencoder EASE^R [6] as the individual recommender, while the mean of individual user profiles followed by the EASE^R's recommendation (i.e., AVG strategy [1]) served as the synchronous GRS baseline.

Finally, at the end of each iteration, the framework controls a convergence criterion, i.e., whether there is a sufficient support for some item: $\exists i : r_{pos}(G,i)^{[it]} \geq |G| - \tau_{neg}$. In such a case, the framework signals a *match* and outputs the item(s) fulfilling the criterion. In production settings, users may opt to continue with the process anyway or accept one of the suggestions as their final decision.

4. Evaluation

In the evaluation, we focused on how quickly a group can reach a match, using different variants of the framework. In particular, we used Round of First Consensus (*RFC*) metric: $RFC_G = min(it) : \exists i : r_{pos}(G,i)^{[it]} \geq |G| - \tau_{neg}$. We compared several instances falling into three high-level categories: *synchronous*, *semi-asynchronous*, and *hybrid*. In one iteration, *synchronous* systems provide the same recommendations to all users. In the *semi-asynchronous* setting, each user receives a personalized

[5]This allows us to treat standard synchronous GRS as a special setting of the semi-asynchronous recommendation framework, where the general recommender is instantiated via some group recommender considering all group members, while $t^{[it]} = 0$.

batch that may include redistributed items previously evaluated by other group members. The *hybrid* setting combines both approaches: a synchronous first iteration (to enable fast convergence) followed by semi-asynchronous subsequent iterations. Individual variants within each family differ further in the choice of the general recommender, the use of feedback from previous iterations, and the scheduling of the redistribution parameter $t^{[it]}$. Table 1 summarizes all evaluated variants.

4.1. Dataset and Groups Construction

We conducted the evaluation in a realistic offline simulation setting using the MovieLens 32M dataset [7], pre-processed similarly to previous works [8, 9]. In particular, we retained only users with at least 50 positive ratings ($r(u,i) > 4$) and items with at least 20 positive interactions, resulting in 89K users, 14K items, and 26M ratings. Following [8], we constructed 1000 synthetic groups of size $|G| = 3$ for each of the *random*, *similar*, and *outlier* group types. *Random* groups were created by uniformly sampling users from the dataset. *Similar* groups were composed of users whose pairwise similarities exceeded the 75th percentile of the similarity distribution. Finally, *outlier* groups consist of two mutually similar users (above the 75th percentile) and one user whose similarity to both falls below the 25th percentile.

In all cases, we used user representations generated by the EASE^R algorithm as their embeddings and cosine similarity for group construction. Note that due to the focus on small groups, we use a strict consensus criterion ($\tau_{neg} = 0$), i.e., an item is a match only if all group members agree.

4.2. Simulational Environment

Like other works aiming to determine group decisions based on offline individual-only data, e.g., [10, 11, 12], we face an important dilemma. While the datasets include some user preferences, they do not fully reflect users' true interests in a given decision context. In particular, some relevant items may be missing from the observed interactions (i.e., outside the users' ground truth), while others, despite having positive feedback, may not be relevant in the current situation.

We address this limitation by simulating user feedback using a second pretrained recommendation model (Funk SVD [13]) that estimates user-item relevance $\hat{r}_{SVD}(u,i)$. The predicted scores are mapped to discrete feedback values $\{positive, neutral, negative\}$ as follows. We first identify the per-user mean μ_u and standard deviation σ_u. Then we obtain standardized positive and negative rating thresholds as

$$\begin{aligned} t_{pos} &= \beta + \frac{\bar{r} - \mu_u}{\sigma_u} \\ t_{neg} &= \beta + \frac{\bar{r} - \delta - \mu_u}{\sigma_u} \,, \end{aligned} \tag{3}$$

where $\bar{r} = 3.5$ is a positive rating threshold, $\delta = 1.0$ is a separation threshold determining the spread of the neutral feedback, and β is a population sentiment bias, denoting the general tendency of the population towards providing positive or negative feedback. Then the estimated rating is standardized as well $\hat{z}_{u,i} = (\hat{r}_{SVD}(u,i) - \mu_u)/\sigma_u$ and finally, the feedback probabilities p_{pos}, p_{neutr}, and p_{neg} are estimated through a sigmoid function σ:

$$\begin{aligned} p_{pos} &= \sigma\big((\hat{z}_{u,i} - t_{pos})/s\big) \\ p_{neg} &= \sigma\big((t_{neg} - \hat{z}_{u,i})/s\big) \\ p_{neutr} &= 1 - p_{pos} - p_{neg} \,, \end{aligned} \tag{4}$$

where $s = 0.7$ denotes the skewness of the estimation. The final vote is sampled from the resulting distribution. Note that with such a definition, we introduce stochasticity into the decision process, through the ratings standardization we adapt to per-user positivity variations, and through positivity threshold $\bar{r}$, separation threshold δ, sigmoid skewness s, and population bias β we can effectively tune the simulation to represent realistic groups. Note that in the current experiments, we only varied the population sentiment bias $\beta \in \{0, 1, 2\}$, where higher values indicate groups that are, in general, less

Table 2
Overall comparison across population bias and recommendation window size. The table reports the average number of iterations until the first group consensus (*RFC*) for different recommendation window sizes ω and population bias values $\beta \in \{0.0, 1.0, 2.0\}$. Lower values indicate faster consensus. The best results per setting are **in bold**, while the second-best are underlined.

	$\omega = 1$			$\omega = 3$			$\omega = 5$			$\omega = 10$		
Algorithm | $\beta =$	0.0	1.0	2.0	0.0	1.0	2.0	0.0	1.0	2.0	0.0	1.0	2.0
Hybrid-Ind	3.08	9.94	27.07	1.57	**3.89**	9.95	1.37	**2.78**	6.41	1.16	1.85	3.74
Hybrid-Grp-EMA	3.11	11.12	29.55	**1.47**	4.01	10.12	1.37	2.91	6.21	1.12	1.86	3.43
Async-Static-Grp	3.15	11.62	29.20	1.47	4.26	9.76	1.38	2.97	6.29	1.13	**1.80**	3.36
Async-Static-Ind	3.09	9.53	28.54	2.49	4.25	9.82	2.12	2.99	**6.06**	1.92	2.29	3.63
Async-Dyn-Ind	3.45	**8.81**	**25.84**	2.25	3.90	**9.56**	2.09	2.92	6.07	1.91	2.31	3.68
Sync	3.08	11.66	28.52	1.47	4.17	10.27	1.36	3.00	6.24	**1.11**	1.83	**3.29**
Sync-EMA	**3.06**	10.88	29.89	1.54	4.19	10.50	**1.36**	2.91	6.08	1.13	1.85	3.45

prone to provide positive feedback (corresponding to 1 or 2 standard deviations). Using the estimated user feedback, standard convergence criteria are applied. If the group does not converge even after each user rates 100 items, the simulation stops.

4.3. Results

Table 2 depicts the overall results for varying population sentiment β and the volume of recommended items per iteration ω. Unsurprisingly, all recommenders tend to achieve better *RFC* with increasing ω, but require more iterations to converge when the overall population sentiment is negative (higher β). We can see that the synchronous baselines are competitive when the population is more permissive and when there are fewer synchronization checkpoints, where the other algorithms may benefit. In these regimes, showing the same items to all users is often sufficient for reaching a fast agreement. At the same time, semi-asynchronous and hybrid variants become particularly competitive with higher β values, where the ability to personalize recommendations and redistribute promising items helps guide the group towards consensus. This trend is especially visible for smaller and medium-sized windows ($\omega = 1, 3, 5$). We also observed that in these stricter settings, semi-asynchronous and hybrid approaches achieve approximately 20% higher convergence rates than the synchronous baselines.

Among the proposed methods, the dynamic semi-asynchronous variant (*Async-Dyn-Ind*) performs particularly well under stricter population settings, suggesting that adaptive control of the exploration–exploitation balance may be beneficial when agreement is harder to achieve. Both hybrid variants also perform well, illustrating the benefits of a combined approach. For $\omega = 1$, each iteration corresponds to a single recommended item. Hence, RFC directly corresponds to the number of evaluated items until the first match, and its values are generally larger. Nevertheless, most semi-asynchronous variants remain competitive, suggesting that the redistribution mechanism itself helps guide the group towards consensus even in this constrained setting. We also focused on the impact of different group types. Nevertheless, differences between *random*, *similar*, and *outlier* groups were negligible, well below the impact of other evaluation settings and the individual recommending algorithms, and showed no clear pattern. For the sake of space, detailed results for different group types are available in the online repository.

5. Conclusions and Limitations

We revisited the assumption of synchronicity in group recommender systems and introduced the semi-asynchronous GRS task, where users receive individualized recommendations while contributing to a shared group decision. To support this scenario, we proposed a framework combining personalized candidate generation with redistribution of promising items across group members. Our results show

that asynchronous and hybrid strategies can match or outperform synchronous baselines in terms of time to consensus, particularly in stricter settings where agreement is harder to achieve.

Despite the promising initial results, our current work has several limitations, particularly the limited breadth of evaluation and reliance on simulated groups and feedback. Evaluating the proposed methods with respect to, e.g., groups of different sizes, different group construction methodologies, or different simulation environments and settings, is a relatively straightforward extension, and we plan to address it in the near future. On the other hand, reliance on simulated data is more challenging to address in itself. To overcome this obstacle, we developed a mobile application (see [14]) that instantiates the semi-asynchronous matching, and we plan to use it as a basis for a series of user studies. These should not only provide results of their own, but also help further tune the proposed simulation evaluation framework to better reflect the behavior of organic groups.

Acknowledgments

This paper has been supported by the Czech Science Foundation (GAČR) project 25-16785S.

Declaration on Generative AI

During the preparation of this work, the author(s) used GPT-4 and Grammarly for grammar and spelling checks. After using these services, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] J. Masthoff, A. Delić, Group recommender systems: Beyond preference aggregation, in: Recommender Systems Handbook, Springer, 2022, pp. 381–420.

[2] K. McCarthy, M. Salamó, L. Coyle, L. McGinty, B. Smyth, P. Nixon, Cats: A synchronous approach to collaborative group recommendation, The Florida AI Research Society (2006).

[3] A. Delic, H. Emamgholizadeh, F. Ricci, Charm: A group recommender chatbot, in: UMAP '23 Adjunct, ACM, 2023. doi:10.1145/3563359.3597388.

[4] T. N. Nguyen, F. Ricci, Dynamic elicitation of user preferences in a chat-based group recommender system, in: SAC '17, ACM, 2017, p. 1685–1692. doi:10.1145/3019612.3019764.

[5] G. Marchionini, Exploratory search: from finding to understanding, Commun. ACM 49 (2006) 41–46. URL: https://doi.org/10.1145/1121949.1121979. doi:10.1145/1121949.1121979.

[6] H. Steck, Embarrassingly shallow autoencoders for sparse data, in: WWW '19, ACM, New York, NY, USA, 2019, p. 3251–3257. URL: https://doi.org/10.1145/3308558.3313710. doi:10.1145/3308558.3313710.

[7] F. M. Harper, J. A. Konstan, The MovieLens datasets: History and context, ACM Trans. Interact. Intell. Syst. (2015). doi:10.1145/2827872.

[8] P. Dokoupil, L. Peska, The effect of similarity metric and group size on outlier selection & satisfaction in group recommender systems, in: Adjunct Proceedings of the 31st ACM Conference on User Modeling, Adaptation and Personalization, UMAP '23 Adjunct, Association for Computing Machinery, New York, NY, USA, 2023, p. 296–301. URL: https://doi.org/10.1145/3563359.3597386. doi:10.1145/3563359.3597386.

[9] P. Dokoupil, L. Peska, Long-term fairness in sequential group recommendations, Knowledge and Information Systems 68 (2026) 37. URL: https://doi.org/10.1007/s10115-025-02642-9. doi:10.1007/s10115-025-02642-9.

[10] O. Kaššák, M. Kompan, M. Bieliková, Personalized hybrid recommendation for group of users: Top-n multimedia recommender, Information Processing Management 52 (2016) 459–477. URL: https://www.sciencedirect.com/science/article/pii/S0306457315001211. doi:https://doi.org/10.1016/j.ipm.2015.10.001.

[11] Z. Huang, X. Xu, H. Zhu, M. Zhou, An efficient group recommendation model with multiattention-based neural networks, IEEE Transactions on Neural Networks and Learning Systems 31 (2020) 4461–4474. doi:10.1109/TNNLS.2019.2955567.

[12] L. Baltrunas, T. Makcinskas, F. Ricci, Group recommendations with rank aggregation and collaborative filtering, in: Proceedings of the Fourth ACM Conference on Recommender Systems, RecSys '10, Association for Computing Machinery, New York, NY, USA, 2010, p. 119–126. URL: https://doi.org/10.1145/1864708.1864733. doi:10.1145/1864708.1864733.

[13] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (2009) 30–37. doi:10.1109/MC.2009.263.

[14] O. Kříž, P. Dokoupil, L. Peška, Match-it: Group recommendations in the age of mobile applications, in: Proceedings of the 34th ACM Conference on User Modeling, Adaptation and Personalization (UMAP '26), ACM, New York, NY, USA, 2026, pp. 1–3. URL: https://doi.org/10.1145/3774935.3812728. doi:10.1145/3774935.3812728.